Web Programming

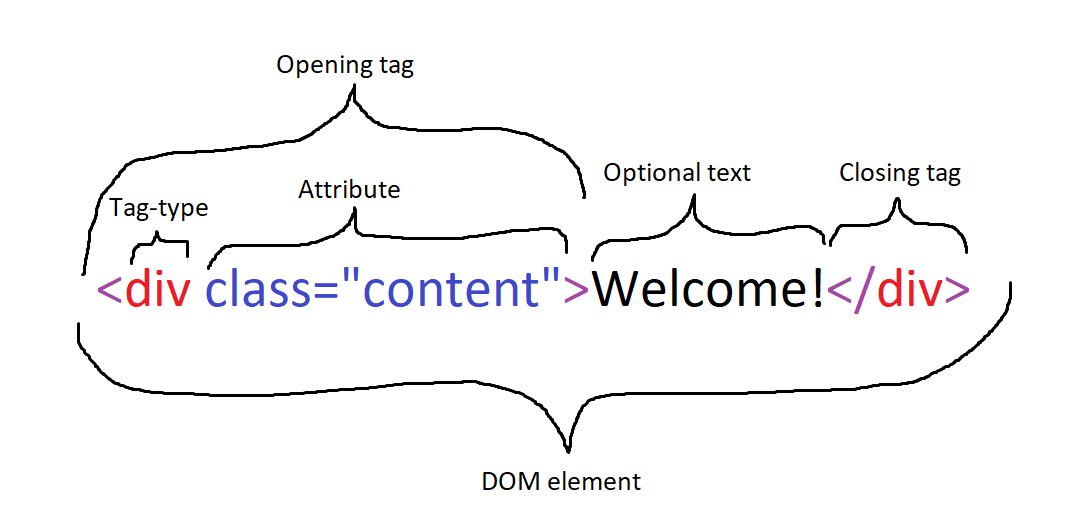
**Introduction**

Web programming is currently one of the largest areas of computer science. There is so much to cover within this field that I will most likely be splitting it into 2 or 3 parts. Our central focus at the moment is front end web development. This area encapsulates all of the user interface and important components that lay at the heart of what websites are. Within front end web development, we typically use 3 core languages. These are often referred to as “the trinity of web development”: HTML, CSS, and JavaScript. Backends are used for processing information (usually from a database). Typically, the backend is written in either PHP, Python, .Net (C#), or some other language that has extensive libraries. Popular databases as of 2020 include MySQL, PostgreSQL, MongoDB, Oracle, etc. We’ve gone over database already, so whenever it’s brought up, this guide will assume that you are familiar with SQL syntax. If you don’t quite remember, I assure you that it will come back to you soon enough. Let’s begin with HTML how about it.

**Hypertext Markup Language (HTML)**

HTML, an abbreviation for hypertext markup language, is in fact, not a programming language, but rather, a convention used for formatting. There are no compilers within websites, otherwise the web would be very slow. The process is quite complex, so I’m not sure that now is a good time to go into detail, but essentially, every browser uses what we call an engine, similar to how games run from engines, such as the unreal engine, unity engine, etc. Chrome runs on the famous/infamous “Chromium Engine” (2 billion lines of code as they claim which is probably not something to boast about) as do many browsers and other applications like Steam. I believe that the official name for chrome’s engine is “Blink”. On the other hand, Firefox uses the Gecko engine. Anyways, these engines are responsible for many, many things, although their main purpose is to interpret HTML into DOM elements. DOM is an important and recurring theme within web development. It stands for Document Object Model; a system of nodes that have parent/child relationships and are able to inherit attributes for size, placement, font-type, etc. Altering these attributes is where CSS comes into play, which we will get to soon enough.

HTML uses tags to “tokenize” predefined DOM elements. In other words, it uses the triangular braces (<>) in association with keywords eg, a, p, div, head, body, etc. to have the engine place specific DOM elements in a particular order. That’s sort of a strange thing about HTML, the placement of elements on your screen are dependent on their placement in your .html file. Within every tag, you can have what are called “attributes” which are key:value pairs that can either alter the element or provide metadata to the element. Think of attributes as additional options similar to Linux commands. They always proceed the tag’s element and are always placed within the first set of braces. Overall, here is the terminology for the contents of an html element:



As you can see here, we create a div tag, short for division, and set the optional class attribute = content. We also place some optional text, which the tag-type and/or attributes may affect in some capacity. Finally, we close the tag with to triangle braces, a backslash, and the type of tag we’re closing. Web has the ability to be very overwhelming with the sheer amount of stuff that you need to know. Luckily, so long as you can vaguely remember the name of a tag or attribute, a quick Google search tends to bring up a good example quickly.

**Charsets**

Something which I’m sure you’ve seen a lot of by now is the term “UTF-8”. UTF-8 is one of the North American character encoding schemes. By this point, you should at least know what the terms ASCII and Unicode mean. ASCII stands for American Standard Code for Information Interchange. It is a table of 128 printable characters from the English charset (well in actuality, only 95 are truly printable). That is to say 128 of the standard characters you might find in any English text. This was not good enough however, as there are many instances where we in America may still want to type in other languages or use unique symbols such as Greek ones for math, or perhaps some currency like pesos or yen. It was then that Unicode was born, with a whopping 143,859 characters! Naturally, this is too overwhelming for most programmers, thus charsets were created to specify which region of Unicode should be used. If we need characters from another language, we must use that charset. HTML 4’s default charset (keep in mind, we’re currently on HTML 5 as of 2021) was ISO-8859-1, and it also had support for UTF-8. This charset supported 256 characters. Window’s first charset was called ANSI (Windows-1252), and it is identical to ISO-8859-1, only it has an additional 32 characters. UTF-8 which we now use in HTML 5 supports most characters and symbols in the world making it very practical for websites. Those are pretty much the 4 North American charsets, though others exist within other countries such as ISO-8859-8 for Hebrew, SHIFT\_JIS for Japan, etc.